

Attorney Docket No. 18897-003001 / 1150-111US

Amendment to the Claims:

This listing of claims replaces all prior versions, and listings, of claims in the application:

1. (Currently amended) A method for dispatching a burst of test packets onto a network, the method comprising:

- generating a plurality of test packets;
- forwarding to an I/O completion port a request that the test packets be dispatched; [[and,]]
- dispatching the test packets onto the network using the I/O completion port;
- measuring departure time of each of the test packets; and
- measuring return time of each of the test packets.

2. (Currently amended) The method of claim 1 wherein the packets are forwarded to the I/O completion port asynchronously[[:]]\_

3. (Original) The method of claim 1 wherein forwarding the test packets to the I/O completion port is performed by a user mode thread during a single time slice.

4. (Original) The method of claim 3 comprising:

- before forwarding the test packets, terminating the current time slice for the user thread; and forwarding the test packets

Attorney Docket No. 18897-003001 / 1150-111US

to the I/O completion port at a start of a next time slice for the user thread.

5. (Original) The method of claim 4 comprising assigning a time-critical priority to the user mode thread.

6. (Original) The method of claim 3 comprising assigning a time-critical priority to the user mode thread.

7. (Original) The method of claim 3 wherein the user mode thread accesses directly buffers in a network interface device.

8. (Original) The method of claim 3 comprising receiving returning dispatched test packets after they have traversed a path in the network and time stamping notifications that the packets have been received.

9. (Original) The method of claim 8 wherein the user mode thread creates in advance, or has created for it in advance, buffers sufficient for receiving all of the returning dispatched test packets.

Attorney Docket No. 18897-003001 / 1150-111US

10. (Original) The method of claim 9 wherein the user mode thread uses a hardware counter for time stamping returning packets.

11. (Original) The method of claim 9 comprising maintaining a private heap for packet data, wherein the private heap is accessible to the user mode thread.

12. (Original) The method of claim 11 wherein the private heap comprises standard-size allocation units for storing packets.

13. (Original) The method of claim 12 wherein the standard-size allocation units are of an operating system memory page size.

14. (Original) The method of claim 13 wherein the standard-size allocation units are 4096 bytes.

15. (Original) The method of claim 11 comprising assigning a larger than default process working set size to the user mode thread.

Attorney Docket No. 18897-003001 / 1150-111US

16. (Original) The method of claim 15 wherein the process working set size exceeds 8 Mbytes.

17. (Original) The method of claim 3 wherein the user mode thread accesses directly buffers in a network card from which the test packets are dispatched onto the network.

18. (Original) The method of claim 1 wherein generating the test packets comprises generating a plurality of equal-sized test packets.

19. (Original) The method of claim 1 wherein generating the test packets comprises generating ethernet test packets.

20. (Original) The method of claim 18 wherein generating the test packets comprises generating a plurality of equal-sized test packets wherein each of the test packets has a size in the range of 46 bytes to 1500 bytes.

21. (Original) The method of claim 1 comprising, receiving from the I/O completion port notifications that the packets have been dispatched and time stamping the notifications.

Attorney Docket No. 18897-003001 / 1150-111US

22. (Original) The method of claim 8 wherein receiving the returning dispatched packets comprises passing data for the returning dispatched packets through an I/O completion port associated with a network interface at which the returning dispatched packets are received.

23. (Currently amended) A program product comprising a computer-readable medium carrying computer-readable signals comprising instructions which, when executed by a computer processor, cause the computer processor to execute a method for dispatching a burst of test packets onto a network, the method comprising:

generating a plurality of test packets;

forwarding to an I/O completion port a request that the test packets be dispatched; [[and,]]

dispatching the test packets onto the network using the I/O completion port;

measuring departure time of each of the test packets; and

measuring return time of each of the test packets.

24. (Currently amended) The program product of claim [[18]] 23 wherein the instructions comprise a controller section

Attorney Docket No. 18897-003001 / 1150-111US

and a test handler section wherein the controller section and test handler section each comprise a separate thread.

25. (Currently amended) Apparatus for dispatching bursts of packets onto a computer network, the apparatus comprising:

a computer processor;

a network interface;

a program memory accessible to the processor, the program memory comprising test packet sequencer software comprising a series of instructions executable by the processor under control of an operating system, the instructions, if executed by the processor, causing the processor to:

establish a first I/O completion port;

generate a plurality of test packets;

forward to the first I/O completion port a request that the test packets be dispatched; [[and,]]

dispatch the test packets onto the network by way of the network interface under control of the first I/O completion port;

measure departure time of each of the test packets; and

measure return time of each of the test packets.

Attorney's Docket No.: 18897-003001/1150-111US

26. (Original) The apparatus of claim 25 wherein the test packet sequencer software comprises a test controller layer associated with a second I/O completion port and a command controller layer associated with the first I/O completion port, wherein the test controller layer is configured to pass commands to the command controller layer by way of the first I/O completion port and the command controller layer is configured to pass raw data to the test controller layer by way of the second I/O completion port.